

Efficient Inference for Flow Matching via Unified Path CFG

Kosuke Ukita¹[\[0009-0008-3325-1363\]](mailto:ukita.kosuke299@mail.kyutech.jp) and Tsuyoshi Okita²[\[0000-0002-1286-5496\]](mailto:tsuyoshi@ai.kyutech.ac.jp)

¹ Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, JAPAN
ukita.kosuke299@mail.kyutech.jp

² Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, JAPAN
tsuyoshi@ai.kyutech.ac.jp

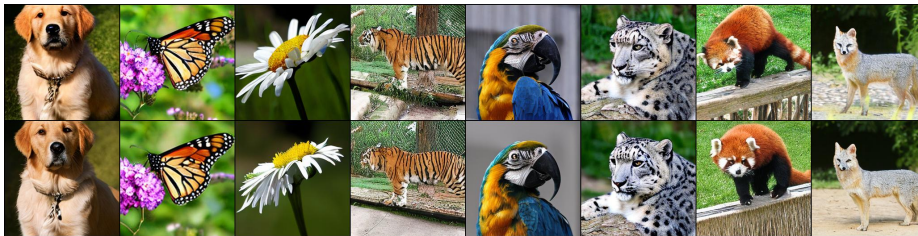


Fig. 1. Qualitative comparison on ImageNet-1k. (Top: Baseline; Bottom :Ours)
Using fixed seeds, our method maintains visual fidelity despite a $2\times$ speedup.

Abstract. Classifier-Free Guidance (CFG) is indispensable for high-fidelity image generation in generative models such as Flow Matching. However, CFG suffers from a severe drawback: it necessitates two forward passes—conditional and unconditional—per step, doubling the inference computational cost. In this paper, we propose CFG with Unified Path (cfgUP) to halve this inference cost. Built on the hypothesis that the internal representations of a conditional path contain sufficient information to reconstruct unconditional predictions, we introduce a lightweight adapter—comprising less than 1% of total parameters—attached to a frozen backbone model. This enables the computation of the CFG term within a single forward pass. Evaluations on ImageNet demonstrate that cfgUP effectively doubles inference speed while maintaining generation quality (FID). Furthermore, our method requires no retraining of the backbone model and can be trained efficiently on a single GPU, significantly enhancing the practicality of large-scale generative models. The implementation code is available at <https://github.com/Okita-Laboratory/cfgUP>.

Keywords: Flow Matching · Efficient Inference · Classifier-Free Guidance.

1 Introduction

Recent deep generative models directly regressing velocity fields between data and noise, such as Flow Matching [14, 1] and Rectified Flow [15], offer highly efficient generation via deterministic ordinary differential equations (ODEs). However, conditional generation in these models faces a critical challenge: samples

often converge to the average of diverse modes, causing blurriness and feature loss. This phenomenon is more pronounced than in diffusion models [26], which mitigate averaging through stochastic noise injection. In contrast, Flow Matching’s deterministic, linear trajectories strictly follow the mean-seeking velocity field, directly leading to averaged outputs.

To resolve mode averaging and ensure conditional fidelity, Classifier-Free Guidance (CFG) [8] is indispensable. However, CFG entails a severe trade-off: it doubles the Number of Function Evaluations (NFE) by requiring two forward passes (conditional and unconditional) per timestep. While methods like InstaFlow [16] reduce total sampling steps to near one, decreasing the per-step inference cost remains an orthogonal challenge. CFG distillation [18] addresses this, but retraining large backbones like SiT [17] is computationally prohibitive and sacrifices the flexible inference-time adjustment of the CFG scale [10].

In this study, we propose CFG with Unified Path (cfgUP), a novel framework that preserves CFG’s mode separation in Flow Matching while dramatically reducing computational costs. Our core hypothesis is that a conditional prediction’s internal representation inherently contains sufficient information to reconstruct its unconditional prediction³. Specifically, we attach a lightweight adapter—comprising less than 1% of total parameters—to a completely frozen pre-trained backbone. This adapter instantly estimates the unconditional prediction from the conditional path’s final features, enabling CFG computation within a single forward pass.

Our main contributions of this work are as follows:

- We propose cfgUP, which doubles Flow Matching inference speed by estimating the unconditional prediction and computing CFG in a single pass.
- We demonstrate on ImageNet that cfgUP successfully mitigates mode averaging, achieving generation quality comparable to standard CFG.
- We show cfgUP is highly practical: it requires no backbone retraining and uses an ultra-lightweight adapter trainable efficiently on a single GPU.

2 Related Works

Our work relates to three main areas: CFG distillation, adapter-based efficiency, and mode separation in Flow Matching.

While CFG [8] is the de facto standard for improving sample fidelity, its requirement of two forward passes doubles the inference cost. To mitigate this, Guidance Distillation [18] and guidance-distilled models (e.g., FLUX.1⁴) have been proposed. However, these methods fix the guidance scale w during training, completely losing the flexibility to dynamically adjust the diversity-fidelity trade-off during inference.

To avoid costly retraining, adapter-based methods that add lightweight modules to frozen backbones have gained attention. For instance, Plug-and-Play Diffusion Distillation [10] and Adapter Guidance Distillation [11] use adapters to

³ The theoretical justification for this hypothesis is detailed in **Appendix 1**.

⁴ <https://huggingface.co/black-forest-labs/FLUX.1-dev>

directly mimic the guided output. However, learning this complex manifold requires massive adapters ($\approx 40\%$ of the backbone); employing a tiny $\approx 1\%$ adapter severely degrades the FID [10]. Furthermore, these methods still lack inference-time scale flexibility. In contrast, cfgUP uniquely separates and reconstructs the unconditional component from the conditional path’s hidden layers. By targeting only the unconditional prediction, cfgUP achieves high accuracy with an ultra-lightweight adapter (0.3%–0.9%) and fully retains dynamic CFG scale control, clearly distinguishing it from prior work.

To address the mode averaging problem specific to conditional Flow Matching, Δ FM [26] introduces an in-batch contrastive loss during pre-training to enforce mode separation. However, this strong regularization distorts the learned velocity fields, compromising the inherent linear trajectories of Flow Matching. This curvature increases numerical integration errors, hindering the use of fast few-step ODE solvers. By completely freezing the pre-trained weights, cfgUP preserves these linear trajectories and compatibility with fast solvers, while still reaping the powerful mode separation benefits of CFG.

3 Background

3.1 Conditional Flow Matching

Flow Matching is a generative framework that learns a continuous probability flow transforming a prior distribution $p(x_0)$ to a data distribution $q(x_1)$. Unlike diffusion models, it models the generation process using deterministic ODEs: $dx_t = v_t(x_t)dt$, $x_0 \sim p(x_0)$ where v_t is the velocity field. For conditional generation (e.g., with a class label c), the model is trained to approximate a conditional velocity field $v_\theta(x_t, t, c)$. The objective function $\mathcal{L}_{CFM}$ minimizes the mean squared error against a target velocity field $u_t(x_t|x_1, x_0)$ (**Equation (1)**).

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t, q(x_1), p(x_0)} [\|v_\theta(x_t, t, c) - u_t(x_t|x_1, x_0)\|^2] \quad (1)$$

Typically, the target is defined as $u_t(x_t|x_1, x_0) = x_1 - x_0$. This linear trajectory enables Flow Matching to perform sampling with relatively few steps.

3.2 Classifier-Free Guidance in Flow Matching

During sampling, CFG is applied to enhance fidelity to the condition c . It computes a modified velocity field $\tilde{v}_\theta$ using both the conditional velocity field $v_\theta(x_t, c)$ and the unconditional velocity field $v_\theta(x_t, \emptyset)$ (**Equation (2)**).

$$\tilde{v}_\theta(x_t, c) = v_\theta(x_t, \emptyset) + s \cdot (v_\theta(x_t, c) - v_\theta(x_t, \emptyset)) \quad (2)$$

where $s > 1$ is the guidance scale. The unconditional prediction $v_\theta(x_t, \emptyset)$ acts as a baseline; extrapolating the difference $(v_\theta(x_t, c) - v_\theta(x_t, \emptyset))$ strongly guides the trajectory toward the conditional mode. However, this process presents a significant computational bottleneck. Calculating **Equation (2)** requires evaluating the network twice per sampling step (**Equation (3)**).

$$\text{NFE}_{CFG} = 2 \times \text{NFE}_{Base} \quad (3)$$

These dual forward passes double the inference latency. Our goal is to eliminate this redundancy and approximate **Equation (2)** within a single forward pass.

4 Proposed Methods

In **Section 4**, we detail our proposed method, CFG with Unified Path (cfgUP), designed to halve the inference cost of CFG in Flow Matching.

While standard CFG requires two independent forward passes, $v_\theta(x_t, c)$ and $v_\theta(x_t, \emptyset)$, we hypothesize that the internal representation of the conditional path retains not only condition-specific information but also the global structural information necessary for unconditional generation. We validate the theoretical and empirical grounds supporting this hypothesis in **Appendix 1**. Therefore, by appropriately transforming the conditional path’s internal representation, we can approximate and reconstruct the output of the unconditional path. Based on this hypothesis, cfgUP adopts a Unified Path framework: it executes a single “conditional path” and simultaneously infers the “unconditional prediction” via a lightweight adapter branching from the intermediate layers, effectively halving the NFE (**Fig. 2**).

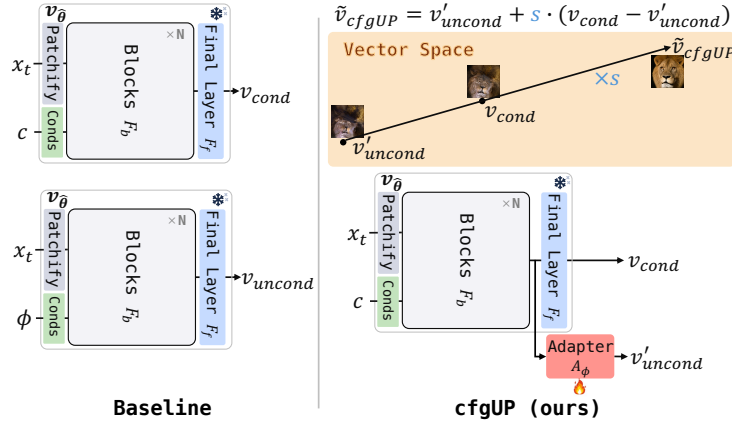


Fig. 2. Overview of cfgUP. Unlike the baseline (left) which requires two full forward passes, our method (right) estimates the unconditional term v'_{uncond} via a lightweight adapter A_ϕ , effectively halving the computational cost.

4.1 Architecture

The proposed architecture comprises a frozen backbone model v_θ and a newly added, trainable lightweight adapter A_ϕ . First, we functionally decompose the backbone model (e.g., SiT [17]) into two components: Blocks⁵ (F_b) and a Final Layer⁶ (F_f). Into this configuration, we introduce the adapter A_ϕ operating in parallel with F_f . The role of this adapter is to transform and remove the influence of condition c from the conditional features h , generating an approximation of the unconditional prediction $v'_{uncond} = A_\phi(h, c)$.

⁵ A set of Transformer blocks that take x_t and condition c to output $h = F_b(x_t, c)$.

⁶ The final layer that projects the features h to the final velocity field $v_{cond} = F_f(h, c)$.

4.2 Inference

During inference, the final guided velocity field $\tilde{v}_{cfgUP}$ is derived within a single pass using the backbone $v_{\hat{\theta}}$ and the trained adapter $A_{\hat{\phi}}$ (**Equation (4)**).

$$\tilde{v}_{cfgUP}(x_t, c) = s \cdot F_f(h, c) + (1 - s) \cdot A_{\hat{\phi}}(h, c) \quad (4)$$

Crucially, both v_{cond} and v'_{uncond} share the intermediate features h output by the Blocks F_b , which dominate the computational cost. Since the adapter A_{ϕ} consists of a small multi-layer perceptron accounting for only 0.39%–0.94% of the total backbone parameters, the additional computational overhead is essentially negligible. A comparison of the inference algorithms for standard CFG and cfgUP is shown in **Algorithm 1** and **Algorithm 2**.

Algorithm 1 CFG (Baseline)	Algorithm 2 cfgUP (Ours)
Require: $v_{\hat{\theta}} = \{F_b, F_f\}$	Require: $v_{\hat{\theta}} = \{F_b, F_f\}, A_{\hat{\phi}}$
Require: c, s	Require: c, s
1: Number of Sampling Steps n	1: Number of Sampling Steps n
2: Start noise $x_n \sim N(0, I)$	2: Start noise $x_n \sim N(0, I)$
3: for $t = n \rightarrow 0$ do	3: for $t = n \rightarrow 0$ do
4: $\triangleright$ Path 1 (Cond.)	4: $\triangleright$ Unified Path
5: $h_c \leftarrow F_b(x_t, c)$	5: $h_c \leftarrow F_b(x_t, c)$
6: $v_c \leftarrow F_f(h_c, c)$	6: $v_c \leftarrow F_f(h_c, c)$
7: $\triangleright$ Path 2 (Uncond.)	7: $v'_{unc} \leftarrow A_{\hat{\phi}}(h_c, c)$
8: $h_{unc} \leftarrow F_b(x_t, \emptyset)$	8: $\triangleright$ Guidance
9: $v_{unc} \leftarrow F_f(h_{unc}, \emptyset)$	9: $v'_g \leftarrow v'_{unc} + s \cdot (v_c - v'_{unc})$
10: $\triangleright$ Guidance	10: $x_t \leftarrow \text{Solver}(x_t, t, v'_g)$
11: $v_g \leftarrow v_{unc} + s \cdot (v_c - v_{unc})$	11: end for
12: $x_t \leftarrow \text{Solver}(x_t, t, v_g)$	12: return x_0
13: end for	
14: return x_0	

4.3 Training

The adapter A_{ϕ} is trained using the frozen backbone model $v_{\hat{\theta}}$. The objective is to bring the predicted value v'_{uncond} close to the true unconditional prediction $v_{\hat{\theta}}(x_t, \emptyset)$. Thus, the training loss function $\mathcal{L}_{adapter}$ is defined in **Equation (5)**:

$$\mathcal{L}_{adapter} = \mathbb{E}_{x_t, c} \left[\|A_{\hat{\phi}}(h, c) - v_{\hat{\theta}}(x_t, \emptyset)\|^2 \right] \quad (5)$$

where $h = F_b(x_t, c)$. During training, only the adapter parameters ϕ are updated. By completely freezing the massive backbone model $v_{\hat{\theta}}$, VRAM usage and computation time are drastically reduced, resulting in an overwhelmingly lower deployment cost compared to existing distillation or retraining methods.

5 Experiments and Evaluation

We evaluate cfgUP to validate three core claims: (1) it halves inference cost, effectively doubling generation speed; (2) it maintains generation quality comparable to standard CFG; and (3) it requires minimal training costs.

5.1 Experimental Setup

Dataset & Models: We benchmark on ImageNet 256×256 [23]. To demonstrate low training cost, we also train on a random 10% subset of the ImageNet-1k training set. We adopt the state-of-the-art Transformer-based Flow Matching model, SiT [17], as our frozen backbone. As detailed in **Table 1**, our adapter adds less than 1% to the total parameter count across all model scales.

Implementation Details: The adapter A_ϕ takes the intermediate features immediately preceding the final layer. We train it using the AdamW optimizer (learning rate 10^{-4} , batch size 32) on a single NVIDIA A100 (40GB) GPU. Peak VRAM usage was merely 16.5 GB. Training converges remarkably fast, completing in approximately 10 hours on the 10% subset.

Table 1. Model architecture details. Summary of parameter counts for each model. Our proposed adapter adds minimal parameters ($< 1\%$) regardless of the model size.

Model size / Patch size	Backbone params	Adapter params	Adapter ratio
S/2	32.9M	0.308M	0.93%
B/2	130M	1.20M	0.92%
L/2	458M	2.13M	0.46%
XL/2	675M	2.69M	0.39%

Evaluation Metrics: We measure generation quality using FID-50k [6]. To ensure strict comparability [20], all FID scores are computed against the ImageNet validation set using the official ADM TensorFlow evaluation script [4]. We also report Inception Score (IS) [24], sFID [19], Precision, and Recall [12].

For efficiency, we measure Inference Latency (s/img) and the Number of Function Evaluations (NFE), defined as the execution count of the Transformer Blocks F_b that dominate the overall inference cost⁷. Unless otherwise noted, we employ the adaptive Dopri5 ODE solver [3] for our experiments. To prioritize inference speed, we set both the absolute (atol) and relative (rtol) tolerances to $1e-2$, resulting in an average NFE of 20–25. In contrast, the standard setting for our baseline, SiT, employs the Euler method with 250 steps (equivalent to NFE = 500 when CFG is applied). Because our experimental results are derived using substantially fewer evaluation steps, the baseline method inherently possesses an advantage in generation quality metrics.

5.2 Results

Generation Quality: As shown in **Table 2**, cfgUP achieves an FID of 3.16^8 , significantly outperforming the unguided SiT-XL/2 baseline (FID 9.35). It demonstrates highly competitive performance against heavily optimized diffusion models and GANs. Crucially, cfgUP bridges the efficiency-quality gap, preserving high-fidelity mode separation despite the drastically reduced computational cost.

Inference Efficiency: **Table 3** highlights the exact speedup. The standard SiT setting takes 17.71s per image. Pairing standard CFG with Dopri5 reduces this

⁷ Typically, NFE in the literature refers to NFE_{CFG} (ignoring the dual passes of CFG). In this paper, NFE strictly denotes NFE_{Base} .

⁸ Dopri5 with tighter tolerances (atol = rtol = $1e-4$), prioritizing quality over speed.

Table 2. Benchmarking class-conditional image generation on ImageNet 256×256 . We report FID, IS, sFID, Precision, and Recall. Our model yields competitive results against leading approaches like SiT, DiT and LDM, validating the effectiveness of our proposed sampling strategy.

Class-Conditional ImageNet 256×256					
Model	FID↓	sFID↓	IS↑	Precision↑	Recall↑
BigGAN-deep [2]	6.95	7.36	171.40	.87	.28
StyleGAN-XL [25]	2.30	4.02	265.12	.78	.53
ADM [4]	10.94	6.02	100.98	.69	.63
ADM-G, ADM-U	3.94	6.14	215.84	.83	.53
CDM [7]	4.88	-	158.71	-	-
LDM-8 [22]	15.51	-	79.03	.65	.63
LDM-4	10.56	-	103.49	.71	.62
LDM-4 (cfg=1.25)	3.95	-	178.22	.81	.55
Simple Diffusion (U-Net) [9]	3.76	-	171.60	-	-
DiT-XL/2 [21]	9.62	6.85	121.50	.67	.67
DiT-XL/2 (cfg=1.5)	2.27	4.60	278.24	.83	.57
SiT-XL/2 (ODE) [17]	9.35	6.38	126.06	.67	.68
SiT-XL/2 (SDE)	8.26	6.32	131.65	.68	.67
SiT-XL/2 (cfg=1.5, ODE)	2.15	4.60	158.09	.81	.60
SiT-XL/2 (cfg=1.5, SDE)	2.06	4.49	277.50	.83	.59
SiT-XL/2 (cfgUP=2.0, ODE) (ours)	3.16	6.77	211.46	.80	.58

to 1.85s. By eliminating the redundant forward pass, our cfgUP further halves the required NFE to 20–25, achieving sub-second latency (0.97s). This equates to a nearly $2\times$ speedup over the standard Dopri5 baseline and an $18.3\times$ overall speedup against the official Euler setting.

Table 3. Inference efficiency comparison. cfgUP drastically reduces NFE, enabling an $18.3\times$ speedup and sub-second latency without sacrificing generation quality.

Method	ODE solver	FID ↓	NFE ↓	Latency ↓	Speedup ↑
SiT-XL/2 (cfg=1.5)	Euler	2.15	500	17.71 s	1.0×
SiT-XL/2 (without CFG)	Euler	9.35	250	8.857 s	2.0×
SiT-XL/2 (cfg=1.5)	Dopri5	-	40 - 50	1.855 s	9.5×
SiT-XL/2 cfgUP (ours)	Dopri5	3.47	20 - 25	0.9692 s	18.3×

Ablation Studies: **Fig. 3** shows that guidance scale behavior mirrors standard CFG, with optimal FID peaking around $s \in [1.5, 2.0]$. **Fig. 4** confirms that cfgUP scales predictably with model size, with XL/2 performing best. Most importantly, **Table 4** demonstrates extreme data efficiency: training the adapter on just a 10% subset yields an identical FID to using the full dataset, while slashing training time by 65% (to 9.86 hours). This proves cfgUP’s minimal deployment cost.

Table 4. Data efficiency ablation. Comparison of training on a 10% subset versus the full ImageNet-1k dataset. Remarkably, the subset achieves an identical FID (3.47) while reducing training time by $\sim 65\%$.

SiT-XL/2 (cfgUP)	FID↓	sFID↓	IS↑	Precision↑	Recall↑	Training Time ↓
Subset (10%)	3.47	8.60	221.05	.80	.56	9.86 h
All (100%)	3.47	4.92	227.83	.81	.54	27.66 h

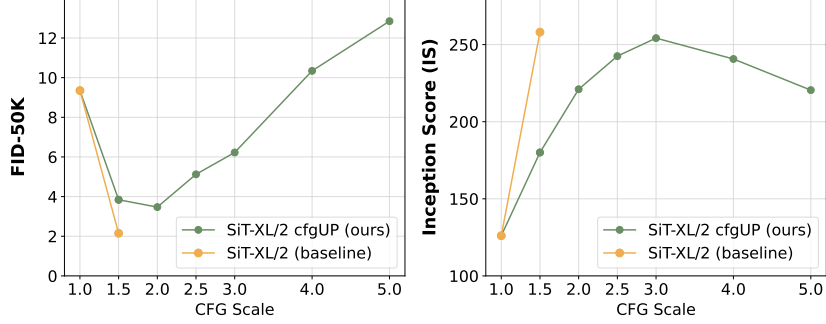


Fig. 3. Ablation study on CFG scale. FID (left) and Inception Score (right) sweeps for SiT-XL/2 with cfgUP. The results demonstrate significant improvements over the unguided baseline at moderate guidance scales.

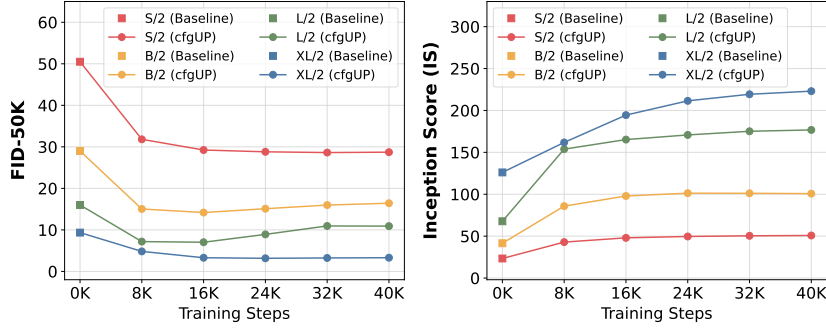


Fig. 4. Impact of training steps and model scaling. FID (left) and Inception Score (right) trajectories for different model sizes. Solid lines denote the training progress of our cfgUP method, initialized from the baseline models (squares at step 0).

Qualitative Results & Generalizability: Visual comparisons (provided in Fig. 1) confirm that cfgUP perfectly preserves the structural layout, textures, and background details of the baseline without introducing visual artifacts, despite the doubled inference speed. Furthermore, to rigorously validate that cfgUP’s mode-separation capabilities generalize beyond ImageNet class-conditioning, we successfully applied our framework to complex Text-to-Image generation using Stable Diffusion 3.5. Detailed methodology and results for this text-conditioned domain are provided in **Appendix 2**.

6 Conclusion

We proposed cfgUP to resolve the efficiency-quality trade-off of Classifier-Free Guidance in Flow Matching. By utilizing a lightweight adapter ($<1\%$ parameters) to reconstruct unconditional terms from conditional representations, our method eliminates the redundant forward pass and halves the NFE. Evaluated on ImageNet, cfgUP achieves an $18.3\times$ speedup over the standard baseline while maintaining high-fidelity generation. Unlike costly distillation methods, cfgUP is a practical, plug-and-play solution that preserves dynamic guidance control for resource-constrained applications.

Appendix

Appendix 1

In this section, we validate the hypothesis that the internal representation of a conditional prediction contains sufficient information to accurately reconstruct its unconditional prediction.

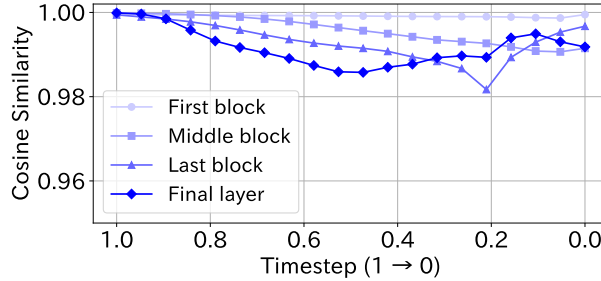


Fig. 5. Cosine similarity between conditional and unconditional features. Across core blocks and all timesteps, the similarity remains exceptionally high (> 0.98), indicating both representations share nearly the same directional subspace.

Fig. 5 illustrates the cosine similarity between conditional and unconditional feature vectors across timesteps. In key layers (First, Middle, Last blocks, and Final layer), the similarity maintains an exceptionally high level (> 0.98), indicating that both representations point in almost the identical direction in the feature space. Although a slight decrease occurs toward the end of generation ($t \rightarrow 0$) as noise removal progresses, the overall similarity remains high. Furthermore, shallower layers exhibit higher similarity, which slightly decreases in deeper layers as class information is refined. These structural characteristics confirm that both representations share the same subspace, facilitating easy prediction by a lightweight adapter.

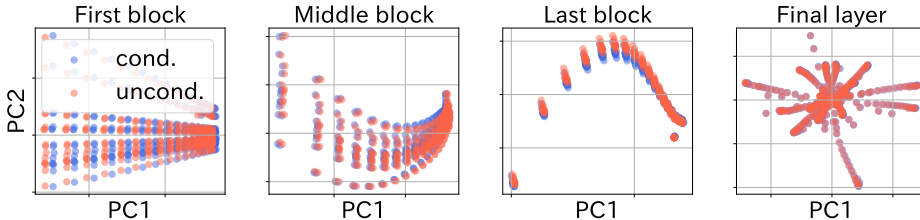


Fig. 6. PCA visualization of features. Conditional and unconditional features overlap heavily on the same manifold across all layers, suggesting that projecting between them requires only minimal transformations.

The 2D PCA visualization of both features across all timesteps (**Fig. 6**) also confirms that their distributions overlap heavily in each layer. This indicates that both representations lie on the exact same manifold within the feature space, visually suggesting that predicting the unconditional representation requires only minor translations along the same manifold, rather than complex projections into different spaces.

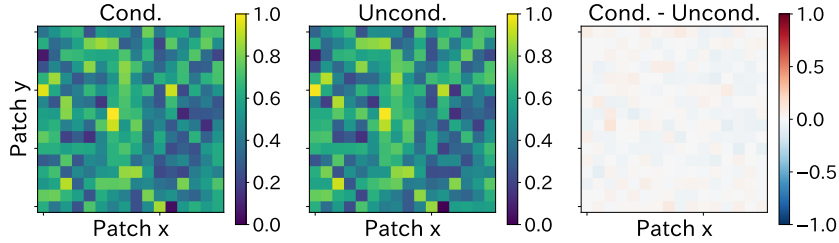


Fig. 7. Attention map comparison at $t = 0.5$: The structural similarities and near-zero differences confirm that extracting unconditional features does not require heavy spatial restructuring.

Finally, **Fig. 7** visualizes the attention maps of the final block output at timestep $t = 0.5$. The conditional and unconditional maps are visually nearly identical, with their difference concentrated around zero. This spatial structural agreement indicates that the difference is limited to minor semantic variations caused by conditioning. Consequently, heavy networks that drastically alter spatial structures are unnecessary, qualitatively verifying that unconditional features can be inferred from conditional ones with extreme ease.

Appendix 2

In this section, we demonstrate that our proposed cfgUP effectively generalizes beyond class-conditional generation to Text-to-Image (T2I) generation tasks, which involve more complex cross-modal conditioning.

We adopted Stable Diffusion 3.5 Medium⁹ [5], a widely used large-scale Flow Matching-based T2I model, as our backbone. Building on the tuning efficiency demonstrated in **Table 4**, we trained the adapter using a minimal dataset of only 10,000 synthesized images generated by the SD3.5 model itself. For evaluation, we randomly sampled 10,000 prompts and corresponding images from the COCO dataset [13]. We evaluated image quality using FID, IS, LPIPS, and CLIP Score. Additionally, we measured Latency [s/img] as an indicator of inference efficiency.

Table 5. Quantitative results on COCO: Comparison between the standard CFG baseline and our cfgUP method using SD3.5 Medium. cfgUP maintains reasonable text-image alignment and perceptual quality while drastically reducing the latency.

Method	FID ↓	IS ↑	LPIPS ↑	CLIP Score ↑	Latency ↓
Baseline (Standard CFG)	23.56	40.16	0.735	31.85	9.23 s
Ours (cfgUP)	37.62	25.49	0.722	29.92	4.77 s

The quantitative results on the COCO dataset are presented in **Table 5**. Although cfgUP shows a drop in image quality metrics such as FID compared to the baseline, it maintains a comparable LPIPS score, indicating that the diversity of generated images is preserved. Furthermore, it achieves a CLIP Score of 29.92, sustaining a reasonable alignment with the text prompts. These results prove that despite the extremely low-cost tuning on just 10K synthesized images, cfgUP successfully reconstructs unconditional predictions in complex T2I conditioning spaces while halving the inference computational cost.

⁹ <https://huggingface.co/stabilityai/stable-diffusion-3.5-medium>



Fig. 8. Qualitative comparison on SD3.5 Medium: Samples generated by the standard CFG (top) and ours (bottom) for the same prompts. Our method successfully preserves the semantic content and overall composition specified by the text conditions.

Fig. 8 provides a qualitative comparison of samples generated by standard CFG and cfgUP. The prompt for the images on the left is “A close up of a clear vase with flowers.”, and for the right is “A set of plush toy teddy bears sitting in a sled.” Despite the halved inference cost, cfgUP visually captures and accurately reproduces the semantics of the specified subjects and the overall image composition. This result strongly supports the practical viability of our Unified Path approach even in large-scale T2I foundation models.

References

1. Albergo, M.S., Vanden-Eijnden, E.: Building normalizing flows with stochastic interpolants. arXiv preprint arXiv:2209.15571 (2022)
2. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096 (2018)
3. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. Advances in neural information processing systems **31** (2018)
4. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021)
5. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first international conference on machine learning (2024)
6. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
7. Ho, J., Saharia, C., Chan, W., Fleet, D.J., Norouzi, M., Salimans, T.: Cascaded diffusion models for high fidelity image generation. Journal of Machine Learning Research **23**(47), 1–33 (2022)
8. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)

9. Hoogeboom, E., Heek, J., Salimans, T.: simple diffusion: End-to-end diffusion for high resolution images. In: International Conference on Machine Learning. pp. 13213–13232. PMLR (2023)
10. Hsiao, Y.T., Khodadadeh, S., Duarte, K., Lin, W.A., Qu, H., Kwon, M., Kalarot, R.: Plug-and-play diffusion distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13743–13752 (2024)
11. Jensen, C.P., Sadat, S.: Efficient distillation of classifier-free guidance using adapters. *Transactions on Machine Learning Research* (2025)
12. Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems* **32** (2019)
13. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
14. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747* (2022)
15. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003* (2022)
16. Liu, X., Zhang, X., Ma, J., Peng, J., et al.: InstafLOW: One step is enough for high-quality diffusion-based text-to-image generation. In: The Twelfth International Conference on Learning Representations (2023)
17. Ma, N., Goldstein, M., Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E., Xie, S.: Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In: European Conference on Computer Vision. pp. 23–40. Springer (2024)
18. Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., Salimans, T.: On distillation of guided diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14297–14306 (2023)
19. Nash, C., Menick, J., Dieleman, S., Battaglia, P.W.: Generating images with sparse representations. *arXiv preprint arXiv:2103.03841* (2021)
20. Parmar, G., Zhang, R., Zhu, J.Y.: On aliased resizing and surprising subtleties in gan evaluation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11410–11420 (2022)
21. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023)
22. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
23. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. *International journal of computer vision* **115**(3), 211–252 (2015)
24. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. *Advances in neural information processing systems* **29** (2016)
25. Sauer, A., Schwarz, K., Geiger, A.: Stylegan-xl: Scaling stylegan to large diverse datasets. In: ACM SIGGRAPH 2022 conference proceedings. pp. 1–10 (2022)
26. Stoica, G., Ramanujan, V., Fan, X., Farhadi, A., Krishna, R., Hoffman, J.: Contrastive flow matching. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 1185–1194 (2025)